

Lecture 2

CP in a Nutshell

Outline

- Introduce notion of equivalence of CSP's
- Provide intuitive introduction to general methods of Constraint Programming
- Introduce basic framework for Constraint Programming
- Illustrate this framework by 2 examples

Projection

- Given: variables $X := x_1, \dots, x_n$ with domains D_1, \dots, D_n

Consider

- $d := (d_1, \dots, d_n) \in D_1 \times \dots \times D_n$

- subsequence $Y := x_{i_1}, \dots, x_{i_j}$ of X

Denote $(d_{i_1}, \dots, d_{i_j})$ by $d[Y]$: **projection** of d on Y

In particular: $d[x_j] = d_j$

- Note: For a CSP

$$\mathcal{P} := \langle C ; x_1 \in D_1, \dots, x_n \in D_n \rangle$$

$(d_1, \dots, d_n) \in D_1 \times \dots \times D_n$ is a solution to \mathcal{P} iff for each constraint C of \mathcal{P} on a sequence of variables Y

$$d[Y] \in C$$

Equivalence of CSP's

- \mathcal{P}_1 and \mathcal{P}_2 are **equivalent** if they have the same set of solutions
- CSP's \mathcal{P}_1 and \mathcal{P}_2 are **equivalent w.r.t. X** iff
$$\{d[X] \mid d \text{ is a solution to } \mathcal{P}_1\} = \{d[X] \mid d \text{ is a solution to } \mathcal{P}_2\}$$
- Union of $\mathcal{P}_1, \dots, \mathcal{P}_m$ is **equivalent w.r.t. X to \mathcal{P}_0** if
$$\{d[X] \mid d \text{ is a solution to } \mathcal{P}_0\} = \bigcup_{i=1}^m \{d[X] \mid d \text{ is a solution to } \mathcal{P}_i\}$$

Solved and Failed CSP's

- C a constraint on variables y_1, \dots, x_k with domains D_1, \dots, D_k (so $C \subseteq D_1 \times \dots \times D_k$):
 C is **solved** if $C = D_1 \times \dots \times D_k$
- CSP is **solved** if
 - all its constraints are solved, and
 - no domain of it is empty
- CSP is **failed** if
 - it contains the false constraint \perp , or
 - some of its domains is empty

CP: Basic Framework

```
procedure solve  
var continue := true  
begin  
    while continue and not happy do  
        Preprocess;  
        Constraint Propagation;  
        if not happy then  
            if Atomic then continue := false  
            else  
                Split; Proceed by Cases  
            end-if  
        end-while  
end
```

Preprocess

Bring to desired syntactic form

- Example: Constraints on reals

Desired syntactic form: no repeated occurrences of a variable

$$ax^7 + bx^5y + cy^{10} = 0$$

$$\rightarrow ax^7 + z + cy^{10} = 0, bx^5y = z$$

Happy

- Found a solution
- Found all solutions
- Found a solved form from which one can generate all solutions
- Determined that no solution exists (inconsistency)
- Found best solution
- Found all best solutions
- Reduced all interval domains to sizes $< \varepsilon$

Atomic and Split

- Check whether CSP is amenable for splitting, or
- whether search ‘under’ this CSP is still needed

Split a domain:

- D finite (Enumeration)

$$\frac{x \in D}{x \in \{a\} \mid x \in D - \{a\}}$$

- D finite (Labeling)

$$\frac{x \in \{a_1, \dots, a_k\}}{x \in \{a_1\} \mid \dots \mid x \in \{a_k\}}$$

- D interval of reals (Bisection)

$$\frac{x \in [a..b]}{x \in \left[a.. \left\lfloor \frac{a+b}{2} \right\rfloor \right] \mid x \in \left[\left\lfloor \frac{a+b}{2} \right\rfloor + 1..b \right]}$$

Split, ctd

Split a **constraint**:

- Disjunctive constraints

$$\frac{C_1 \vee C_2}{C_1 \mid C_2}$$

- Constraints in “compound” form
Example:

$$\frac{|p(\bar{x})|=a}{p(\bar{x})=a \mid p(\bar{x})=-a}$$

Effect of Split

- Each Split replaces current CSP \mathcal{P} by CSP's $\mathcal{P}_1, \dots, \mathcal{P}_n$ such that the union of $\mathcal{P}_1, \dots, \mathcal{P}_n$ is equivalent to \mathcal{P} .

- Example:

Enumeration replaces

$$\langle C ; \mathcal{D}\mathcal{E}, x \in D \rangle$$

by

$$\langle C' ; \mathcal{D}\mathcal{E}, x \in \{a\} \rangle$$

and

$$\langle C'' ; \mathcal{D}\mathcal{E}, x \in D - \{a\} \rangle$$

where C' and C'' are restrictions of the constraints from C to the new domains.

Heuristics

Which

- variable to choose
- value to choose
- constraint to split

Examples:

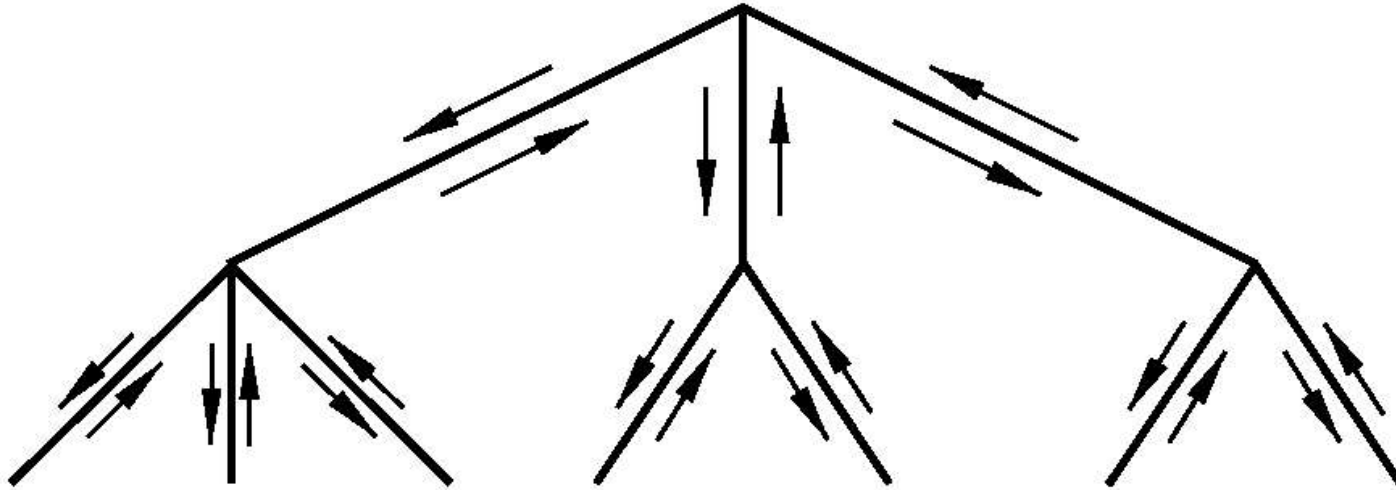
- Select a variable that appears in the largest number of constraints (**most constrained variable**)
- For a domain being an integer interval: select the middle value

Proceed by Cases

Various search techniques

- Backtracking
- Branch and bound
- Can be combined with Constraint Propagation
- Intelligent backtracking

Backtracking

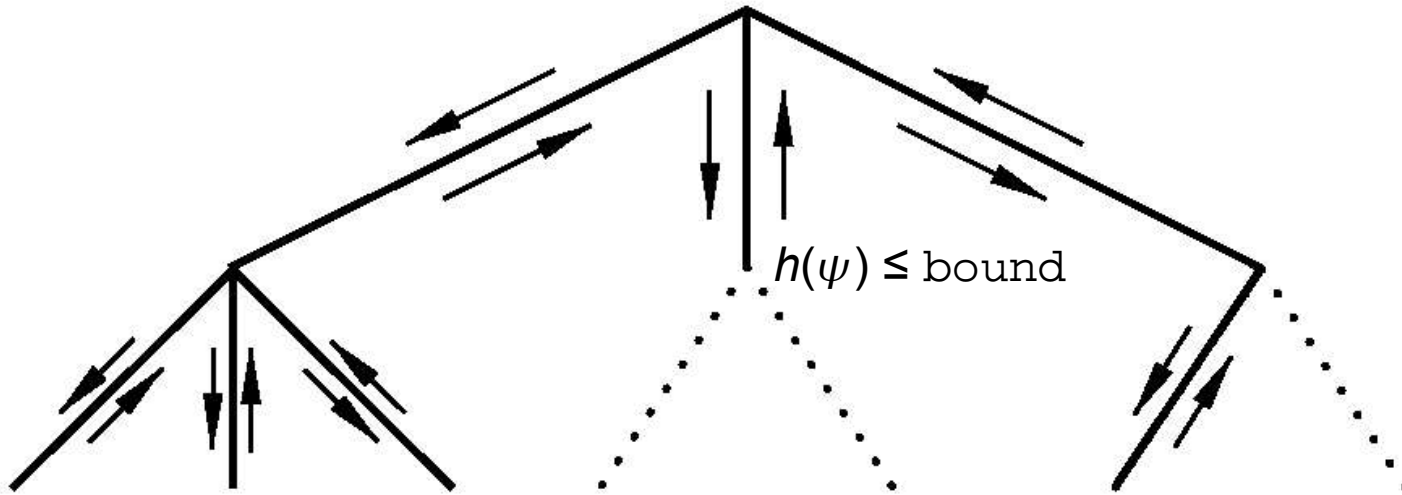


- Nodes generated “on the fly”
- Nodes are CSP's
- Leaves are CSP's that are solved or failed

Branch and Bound

- Modification of backtracking aiming at finding the optimal solution
- Takes into account objective function
- Maintain currently best value of the objective function in variable **bound**
- **bound** initialized to $-\infty$ and updated each time a better solution found
- Used in combination with heuristic function
- Conditions on heuristic function h :
 - If ψ is a direct descendant of ϕ , then
$$h(\psi) \leq h(\phi)$$
 - If ψ is solved CSP with singleton set domains, then
$$\text{obj}(\psi) \leq h(\psi)$$
- h allows us to prune the search tree

Illustration



Constraint Propagation

Replace a CSP by an equivalent one that is “simpler”

Constraint propagation performed by repeatedly reducing

- domains

and/or

- constraints

while maintaining equivalence

Reduce a Domain: Examples

- Projection rule:
Take a constraint C and choose a variable x of it with domain D .
Remove from D all values for x that do not participate in a solution to C .
- Linear inequalities on integers:

$$\frac{\langle x < y; x \in [50..200], y \in [0..100] \rangle}{\langle x < y; x \in [50..99], y \in [51..100] \rangle}$$

Repeated Domain Reduction: Example

Consider

$$\langle x < y, y < z ; x \in [50..200], y \in [0..100], z \in [0..100] \rangle$$

Apply above rule to $x < y$:

$$\langle x < y, y < z ; x \in [50..99], y \in [51..100], z \in [0..100] \rangle$$

Apply it now to $y < z$:

$$\langle x < y, y < z ; x \in [50..99], y \in [51..99], z \in [52..100] \rangle$$

Apply it again to $x < y$:

$$\langle x < y, y < z ; x \in [50..98], y \in [51..99], z \in [52..100] \rangle$$

Reduce Constraints

Usually by introducing new constraints!

- Transitivity of $<$:

$$\frac{\langle x < y, y < z; \mathcal{DE} \rangle}{\langle x < y, y < z, x < z; \mathcal{DE} \rangle}$$

This rule introduces new constraint $x < z$

- Resolution rule:

$$\frac{\langle C_1 \vee L, C_2 \vee \bar{L}; \mathcal{DE} \rangle}{\langle C_1 \vee L, C_2 \vee \bar{L}, C_1 \vee C_2; \mathcal{DE} \rangle}$$

This rule introduces new constraint $C_1 \vee C_2$

Constraint Propagation Algorithms

- Deal with scheduling of atomic reduction steps
- Try to avoid useless applications of atomic reduction steps
- Stopping criterion for general CSP's: a **local consistency** notion

Example:

Local consistency criterion corresponding to the projection rule is **Hyper-arc consistency**:

For every constraint C and every variable x with domain D , each value for x from D participates in a solution to C .

Example: Boolean Constraints

Happy: found all solutions

Desired syntactic form (for preprocessing):

- $x = y$
- $\neg x = y$
- $x \wedge y = z$
- $x \vee y = z$

- Preprocessing:

$$\frac{x \wedge s = z}{x \wedge y = z, s = y}$$

- Constraint propagation:

$$\frac{\langle x \wedge y = z; x \in D_x, y \in D_y, z \in \{1\} \rangle}{\langle ; x \in D_x \cap \{1\}, y \in D_y \cap \{1\}, z \in \{1\} \rangle}$$

(write as $x \wedge y = z, z = 1 \rightarrow x = 1, y = 1$)

Boolean Constraints, ctd

- $x = y, x = 1 \Rightarrow y = 1$
- $x = y, y = 1 \Rightarrow x = 1$
- $x = y, x = 0 \Rightarrow y = 0$
- $x = y, y = 0 \Rightarrow x = 0$

- $x \wedge y = z, x = 1, y = 1 \Rightarrow z = 1$
- $x \wedge y = z, x = 1, z = 0 \Rightarrow y = 0$
- $x \wedge y = z, y = 1, z = 0 \Rightarrow x = 0$
- $x \wedge y = z, x = 0 \Rightarrow z = 0$
- $x \wedge y = z, y = 0 \Rightarrow z = 0$
- $x \wedge y = z, z = 1 \Rightarrow x = 1, y = 1$

- $\neg x = y, x = 1 \Rightarrow y = 0$
- $\neg x = y, x = 0 \Rightarrow y = 1$
- $\neg x = y, y = 1 \Rightarrow x = 0$
- $\neg x = y, y = 0 \Rightarrow x = 1$

- $x \vee y = z, x = 1 \Rightarrow z = 1$
- $x \vee y = z, x = 0, y = 0 \Rightarrow z = 0$
- $x \vee y = z, x = 0, z = 1 \Rightarrow y = 1$
- $x \vee y = z, y = 0, z = 1 \Rightarrow x = 1$
- $x \vee y = z, y = 1 \Rightarrow z = 1$
- $x \vee y = z, z = 0 \Rightarrow x = 0, y = 0$

Boolean Constraints, ctd

Split:

- Choose the most constrained variable
- Apply the labeling rule:

$$\frac{x \in \{0, 1\}}{x \in \{0\} \mid x \in \{1\}}$$

Proceed by cases: backtrack

Example: Polynomial Constraints on Integer Intervals

Domains: integer intervals $[a..b]$

$$[a..b] := \{x \in \mathbb{Z} \mid a \leq x \leq b\}$$

Constraints:

$$s = 0$$

s is a polynomial (possibly in several variables) with integer coefficients

Example:

$$2 \cdot x^5 \cdot y^2 \cdot z^4 + 3 \cdot x \cdot y^3 \cdot z^5 - 4 \cdot x^4 \cdot y^6 \cdot z^2 + 10 = 0$$

Objective function: a polynomial

Example

Find a solution to

$$x^3 + y^2 - z^3 = 0$$

in $[1..1000]$ such that

$$2 \cdot x \cdot y - z$$

is maximal.

Answer:

$$x = 112, y = 832, z = 128$$

Polynomial Constraints on Integer Intervals, ctd

Desired syntactic form:

- $\sum_{i=1}^n a_i x_i = b$
- $x \cdot y = z$

Preprocess:

Use appropriate transformation rules

Example:

$$\frac{\left\langle \sum_{i=1}^n m_i = 0; \mathcal{DE} \right\rangle}{\left\langle \sum_{i=1}^n v_i = 0, m_1 = v_1, \dots, m_n = v_n; \mathcal{DE}, v_1 \in \mathbb{Z}, \dots, v_n \in \mathbb{Z} \right\rangle}$$

where

- some m_i is not of the form ax_i
- v_1, \dots, v_n do not appear in \mathcal{DE}

Happy: found an optimal solution w.r.t. the objective function

Polynomial Constraints on Integer Intervals, ctd

Constraint propagator: uses interval arithmetic

X, Y sets of integers

- addition:

$$X + Y := \{x + y \mid x \in X, y \in Y\}$$

- subtraction:

$$X - Y := \{x - y \mid x \in X, y \in Y\}$$

- multiplication:

$$X \cdot Y := \{x \cdot y \mid x \in X, y \in Y\}$$

- division:

$$X/Y := \{u \in \mathbb{Z} \mid \exists x \in X \exists y \in Y u \cdot y = x\}$$

Interval Arithmetic, ctd

Given: X, Y integer intervals, a an integer

- $X \cap Y, X + Y, X - Y$ are integer intervals
- $X/\{a\}$ is an integer interval
- $X \cdot Y$ does not have to be an integer interval, even if $X = \{a\}$ or $Y = \{a\}$
- X/Y does not have to be an integer interval

Examples:

$$[2..4] + [3..8] = [5..12]$$

$$[3..7] - [1..8] = [-5..6]$$

$$[3..3] \cdot [1..2] = \{3, 6\}$$

$$[3..5]/[-1..2] = \{-5, -4, -3, 2, 3, 4, 5\}$$

$$[-3..5]/[-1..2] = \mathbb{Z}$$

Turning Sets to Intervals

$$\mathit{int}(X) := \begin{cases} \text{smallest int. interval} \supseteq X & \text{if } X \text{ finite} \\ \mathbb{Z} & \text{otherwise} \end{cases}$$

Examples:

$$\mathit{int}([3..3] \cdot [1..2]) = [3..6]$$

$$\mathit{int}([3..5]/[-1..2]) = [-5..5]$$

$$\mathit{int}([-3..5]/[-1..2]) = \mathbb{Z}$$

Rule for Linear Equality

$$\frac{\left\langle \sum_{i=1}^n a_i x_i = b; x_1 \in D_1, \dots, x_n \in D_n \right\rangle}{\left\langle \sum_{i=1}^n a_i x_i = b; \dots, x_j \in D'_j, \dots \right\rangle}$$

where $j \in [1..n]$, and

$$D'_j := D_j \cap \frac{b - \sum_{i \in [1..n] - \{j\}} \text{int}(a_i \cdot D_i)}{a_j}$$

Multiplication Rules

Multiplication 1

$$\frac{\langle x \cdot y = z; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z; x \in D_x, y \in D_y, z \in D_z \cap \text{int}(D_x \cdot D_y) \rangle}$$

Multiplication 2

$$\frac{\langle x \cdot y = z; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z; x \in D_x \cap \text{int}(D_z / D_y), y \in D_y, z \in D_z \rangle}$$

Multiplication 3

$$\frac{\langle x \cdot y = z; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z; x \in D_x, y \in D_y \cap \text{int}(D_z / D_x), z \in D_z \rangle}$$

Effect of Multiplication Rules

Consider

$$\langle x \cdot y = z ; x \in [1..20], y \in [9..11], z \in [155..161] \rangle$$

Using Multiplication Rules we can transform this to

$$\langle x \cdot y = z ; x \in [16..16], y \in [10..10], z \in [160..160] \rangle$$

Polynomial Constraints on Integer Intervals, ctd

Split:

- Choose the variable with the smallest interval domain
- Apply the bisection rule:

$$\frac{x \in [a..b]}{x \in \left[a.. \left\lfloor \frac{a+b}{2} \right\rfloor \right] \mid x \in \left[\left\lfloor \frac{a+b}{2} \right\rfloor + 1..b \right]}$$

where $a < b$

- **Proceed by cases:** branch and bound

More on Interval Arithmetic

Given objective function obj .

obj^+ : extension of obj to function from sets of integers to sets of integers.

Example: $obj(x,y) := x^2 \cdot y - 3x \cdot y^2 + 5$

Then $obj^+(X,Y) = X \cdot X \cdot Y - 3 \cdot X \cdot Y \cdot Y + 5$

Lemma

Consider integer intervals X_1, \dots, X_n

- $obj^+(X_1, \dots, X_n)$ is a finite set of integers

- For all $a_i \in X_i, i \in [1..n]$

$$obj(a_1, \dots, a_n) \in obj^+(X_1, \dots, X_n)$$

- For all $Y_i \subseteq X_i, i \in [1..n]$

$$obj(Y_1, \dots, Y_n) \subseteq obj^+(X_1, \dots, X_n)$$

Heuristic Function

Take

- $\mathcal{P} := \langle C ; x_1 \in D_1, \dots, x_n \in D_n \rangle$, with D_1, \dots, D_n integer intervals
- *obj*: polynomial with variables x_1, \dots, x_n

Define

$$h(\mathcal{P}) := \max(\text{obj}^+(D_1, \dots, D_n))$$

Thanks to the preceding lemma, h satisfies the conditions for the heuristic function (cf. Slide 15).

Objectives

- Introduce notion of equivalence of CSP's
- Provide intuitive introduction to general methods of Constraint Programming
- Introduce a basic framework for Constraint Programming
- Illustrate this framework by 2 examples